

Appl. No. 09/964,998  
Amdt. dated 09/08/2004  
Reply to Office Action of 06/10/2004

REMARKS

Claims 1 - 20 are pending in the present Application. In the above-identified Office Action, the Examiner rejected Claims 1, 6, 11 and 16 under 35 U.S.C. §102(b) as being anticipated by Duggan et al. (US Patent 5,881,289). Claims 2 - 5, 7 - 10, 12 - 15 and 17 - 20 were rejected under 35 U.S.C. §103(a) as being unpatentable over Duggan et al. in view of Farber et al. (US Patent 5,903,760).

Independent Claims 1, 6, 11 and 16 have been amended to overcome the §102(b) rejection. Particularly, Claims 1, 6, 11 and 16 have been amended to include the limitations of previous Claims 3, 8, 13 and 18, respectively. Independent Claims 1, 6, 11 and 16 are further amended to properly claim the invention. Likewise, Claims 2 - 5, 7 - 10, 12 - 15 and 17 - 20 are amended to properly claim the invention. For the reasons stated more fully below, Applicants submit that the claims in the Application are allowable over the applied references. Hence, reconsideration, allowance and passage to issue are respectfully requested.

As stated in the SPECIFICATION, in today's environment, a network may consist of different computer systems running under different operating systems and using different software management utilities. The network is generally managed by a system administrator who ensures that the network is operational and running at its optimum.

To perform this task, the system administrator periodically runs tests and executes management commands on the various systems in the network. When doing so, the

AUS920010904US1

Appl. No. 09/964,998  
Amdt. dated 09/08/2004  
Reply to Office Action of 06/10/2004

system manager has to access each management software utility and run as many commands as there are system management utilities in use in the network. This can be a rather time-consuming endeavor. Hence, a need exists for a simplified mechanism to efficiently manage the different systems in the network. The present invention provides for such a simplified mechanism.

In accordance with the teachings of the invention, a common interface into which commands can be entered is provided. The commands may include a request to start execution of a command, to stop execution of a command or to provide command execution progress status. In any case, a command entered into the common interface is translated into different command structures that may be used by the different software management utilities used by the different computer systems in the network. After being translated into the different command structures, each translated command is dispatched to an appropriate computer system for execution.

Note that the ability of the administrator to request command execution progress status greatly facilitates the task of the administrator. For example, if for any reason execution of a command cannot be completed by a computer system and if command execution progress status was requested, the administrator will be made aware in real-time of such an occurrence. At that time, appropriate actions may be taken.

The invention is set forth in claims of varying scopes of which Claim 1 is illustrative.

AUS920010904US1

Appl. No. 09/964,998

Amdt. dated 09/08/2004

Reply to Office Action of 06/10/2004

1. A method of executing system management commands on computer systems in a network, said computer systems running different system management software utilities having different command structures, said method comprising the steps of:

entering the commands in a common interface, **the commands including requests to start execution of the commands, to stop execution of the commands and to provide command execution progress status to the common interface**, said common interface translating said commands into the different command structures;

dispatching each translated command to an appropriate computer system; and

executing the dispatched command.

(Emphasis added.)

As stated above, independent Claim 1 has been amended to include the limitations of dependent Claim 3, which the Examiner rejected. In rejecting Claim 3, the Examiner admitted that Duggan et al. do not teach that the command includes a command to provide command execution progress status back to the common interface. However, the Examiner stated that Farber et al. teach such limitations. More specifically, the Examiner stated that status flags that are used to determine whether a condition of a conditional instruction is met, as explained by Farber et al., are equivalent to **providing command execution progress status to the common interface**. Applicants respectfully disagree.

Farber et al. purport to teach a method and apparatus for translating a conditional instruction compatible with a first instruction set architecture (ISA) into a conditional instruction compatible with a second ISA. According to the  
AUS920010904US1

Appl. No. 09/964,998  
Amdt. dated 09/08/2004  
Reply to Office Action of 06/10/2004

teachings of Farber et al., when a RISC (Reduced Instruction Set Computer) system is emulating a CISC (Complex Instruction Set Computer) system, the inherent performance advantages of the RISC system may be obviated.

As an example, Farber et al. explain that when a conditional instruction, such as a compare instruction, is processed by a CISC machine, the result is typically based on a previously executed arithmetic instruction. Particularly, two values are compared by subtracting one from the other. A set of status flags is then updated based on the difference of the two values. After doing so, an evaluation of a logical combination of the status flags is undertaken in order to determine whether the condition of the conditional instruction is met. In a CISC machine, the updated status flags are not saved in a memory location but rather in a dedicated register.

A RISC machine, on the other hand, does not have such a dedicated register. Therefore, when emulating a CISC machine, the RISC machine uses a general purpose register to store the updated status flags. Then the RISC system uses a series of instructions provided in the RISC instruction set to evaluate the status flags. However, a RISC system typically includes in its instruction set one conditional instruction to efficiently evaluate a single value generated by a previously executed compare instruction. Since in the emulation, the RISC system does not make use of this one conditional instruction to evaluate the compare instruction, its performance advantage over a CISC system is obviated.

AUS920010904US1

Appl. No. 09/964,998  
Amdt. dated 09/08/2004  
Reply to Office Action of 06/10/2004

Thus, Farber et al. do not teach, show or so much suggest that the limitations of ***providing command execution progress status to the common interface*** as claimed.

Therefore, Applicants submit that Claim 1 and its dependent claims should be allowable. The other independent claims, which all incorporate the above-emboldened-italicized limitations shown in the reproduced Claim 1 above, should be allowable as well. Consequently, reconsideration, allowance and passage to issue are once more respectfully requested.

Respectfully submitted,  
Abdelhadi et al.

By: 

Volel Emile  
Attorney for Applicants  
Registration No. 39,969  
(512) 306-7969

AUS920010904US1

Page 12 of 12